

PRIORITY INFORMATION

This application expressly claims the benefit of the filing date established with U.S. Provisional Application No. 60/133,624 filed May 11, 1999 by Perlin, et al. and commonly assigned to the assignee of the present application.

TECHNICAL FIELD

This invention relates to a class of devices commonly known as smart cards and, in particular, to an interlaced protocol for smart card application development.

BACKGROUND OF THE INVENTION

Today there is increasing use of integrated circuit (IC) cards, colloquially referred to as “smart cards”, in place of, or in addition to, conventional magnetic stripe cards (“mag cards”). A smart card is a thin card embedded with a memory device (volatile and/or non-volatile) and associated programmable or non-programmable logic. Unlike the mag card that merely stores “static” information (e.g., a credit card account number), a smart card can add, delete and otherwise manipulate information stored on the card. Accordingly, smart cards are capable of storing and executing applications to carry out one or more functions within a smart card.

While the physical dimensions and processing features of the smart card give rise to potentially limitless applications, the reality is that smart card applications are typically only developed for large scale markets, e.g., banking, security and transportation applications. One reason for this limited growth lies in the cost associated with smart card application development. There are several reasons why smart card application development is a costly undertaking, not the

1 least of which is the "closed" nature of the smart card and the limited processing,
2 memory and input/output resources of the smart card.

3 A smart card is often referred to as a "closed" system because, for security
4 purposes, a smart card is purposefully designed to not expose its memory,
5 intermediate system states or data and address bus information to external devices.
6 To do so would render it susceptible to unauthorized access (hacking) and fraud.
7 While its closed nature is useful for secure applications such as banking
8 transactions, it makes it difficult to utilize prior art smart cards for development
9 purposes. It is to be appreciated that application development often requires
10 access to memory or bus values, or system state information during intermediate
11 processing steps, access that has been specifically designed out of the smart card.

12 Another encumbrance to the smart card application designer is the limited
13 resources of the smart card. That is, due to the physical and processing constraints
14 placed on the smart card, prior art smart cards do not enjoy any dedicated debug
15 facilities. Aside from the limited processing and memory attributes of a smart
16 card, a smart card typically has but a single, bi-directional input/output (I/O) port.
17 The communication bandwidth of this single I/O port is typically consumed to
18 support execution of the smart card application itself, leaving little to no
19 communication bandwidth to support debug features. Thus, application
20 development using a smart card itself is virtually impossible. Consequently the
21 development of applications for a smart card currently requires the use of an in-
22 circuit emulator (ICE) and an associated, often proprietary software development
23 application.

24 A ICE system is typically comprised of a printed circuit card coupled to a
25 computer system executing a proprietary software development application

associated with the printed circuit card (emulator). The printed circuit card is designed to emulate the functionality of the smart card, while providing additional debug facilities (e.g., I/O ports, memory buffers, address and data lines and the like), thereby providing the developer with the necessary access to adequately debug their applications in development. One limitation of such smart card development systems is that the ICE and proprietary development application are chip-specific. Thus, an emulator for smart card employing a Siemens processor will not work with an emulator employing a Philips or Motorola processor without significant hardware modification. Moreover, the software development application executing on the computer system is also chip-specific, with an associated chip-specific compiler, linker and debugger, and often require that a developer learn the "programming language" of the development tool. Consequently, an application developed on one ICE system cannot be utilized (or directly ported to) a smart card employing a different processor without costly modification.

As a result of each of the foregoing limitations, smart card application development is a costly undertaking, typically performed by the large corporations that stand to profit from the sale of millions of smart cards. History has shown that in order for a new technology to blossom, "grass roots" application development is required. That is, a technology will not truly become a pervasive technology unless and until it is infused with the vitality and creativity of individual programmers and small development companies.

Thus, an improved application development environment is required for smart card applications that is unencumbered by the limitations commonly associated with the prior art. One such solution is presented below.

1
2 SUMMARY OF THE INVENTION

3 This invention concerns an integrated circuit (IC) card, such as a smart card
4 and, more particularly, an interlaced protocol for smart card application
5 development.

6 In accordance with a first aspect of the invention, an IC is presented
7 comprising an input/output (I/O) interface and a smart card development interface
8 (SCDI), coupled to the I/O interface, to receive and identify debug frames
9 interlaced within a normal communication flow between the smart card and an
10 application executing on a host system.

11 According to another aspect of the invention, a computer system is
12 presented comprising a client development interface (CDI), to receive and identify
13 debug frames interlaced within a normal communication flow received from a
14 communicatively coupled integrated circuit (IC) card.

15 According to yet another aspect of the present invention, a protocol
16 enabling smart card application debugging using an IC card is introduced. In
17 particular, a protocol facilitating communication between a host system and an IC
18 card is presented comprising a plurality of application frames and one or more
19 debug frames. The application frames facilitate communication between a host
20 application and one or more smart card resources. The debug frame is interlaced
21 with the application frames comprising normal communication flow to invoke one
22 or more smart card resources. In one embodiment, the host application and the
23 debug application are executing on separate host systems utilizing the resources of
24 an IC card.
25

1 Fig. 5 is a block diagram of an example smart card development interface
2 including a debug filter, according to one aspect of the present invention;

3 Fig. 6 graphically illustrates an example communication flow including a
4 diagram of an example debug frame suitable for use in the application
5 development system of Fig. 1;

6 Fig. 7 is a flow chart of an example method for debugging an IC card
7 application using an interlaced debug protocol, according to one aspect of the
8 present invention; and

9 Fig. 8 illustrates a signaling diagram for an example communication flow
10 between a host system and a smart card utilizing the interlaced debug protocol of
11 the present invention.

12 13 DETAILED DESCRIPTION

14 Example Development System

15 Fig. 1 illustrates an application development system 100 comprising a
16 computer system 102 coupled to smart card 104 in which a user can develop and
17 debug application code for use on a smart card such as, but not limited to, smart
18 card 104. As shown, the development system of Fig. 1 depicts computer system
19 102 coupled to smart card 104 via a card reader 106 and a communication medium
20 108. The communication medium 108 is intended to represent any of a number of
21 typical communication links including, but not limited to, a proprietary data bus,
22 an industry standard data bus, a local area network (LAN), a wide area network
23 (WAN), or a global area network (e.g., the Internet). In this regard, as will be
24 developed more fully below, the innovative application development system 100
25 facilitates development of smart card applications using an actual smart card 104

coupled to one or more computer system(s) (e.g., 102) via a communications network 108 and a card reader 106.

Smart card 104 comprises an innovative smart card development interface (SCDI) 110 coupled to an operating system (OS) 112. According to one aspect of the invention, to be developed more fully below, SCDI 110 receives and identifies debug frames interlaced with "normal" application frames within the normal communication flow between the smart card 104 and an application executing within an application development tool running on computer system 102.

It is to be appreciated that, but for the innovative smart card development interface 110, smart card 104 and its operating system 112 with associated system resources is intended to represent any of a broad range of integrated circuit cards and their operating systems commonly known in the art. That is, any smart card endowed with the innovative smart card development interface 110 is suitable for use within the innovative smart card application development system 100 of Fig. 1.

It is noted that, in addition to the illustrated smart cards, the IC device might be embodied in other forms, such as an electronic wallet, a personal digital assistant, a smart diskette (i.e., an IC-based device having a form factor and memory drive interface to enable insertion into a floppy disk drive), a PC card (formerly PCMCIA card), and the like. Generally, the integrated circuit card 104 is characterized as an electronic device with limited processing capabilities and memory wherein large size number crunching is impractical.

Card reader 106 provides a necessary interface between smart card reader 104 and a computing system such as, e.g., computer system 102. Card readers are typically designed to support any of a number of standardized communication

protocols supported within the smart card community and, in this way, can typically accommodate smart cards adhering to any of the recognized communication standards from any smart card manufacturer. In this regard, card reader 106 is not chip- or card-specific. For purposes of this discussion, card reader 106 includes the necessary hardware and software resources required to support the interlaced debug protocol of the present invention. Consequently, card reader 106 is merely intended to be illustrative of card readers typically known within the art.

Computer system 102 is depicted within Fig. 1 as comprising an innovative client development interface (CDI) 114, a plurality of executable applications 116 including application development tool 118 with a debug environment, and an operating system 120, coupled as shown. The application development tool 118 enables a user to code and debug a smart card application, utilizing a debug environment that generates debug frames. According to one aspect of the present invention, the CDI 114 marshals and interlaces the debug frames with application frames (normal communication flow) generated by the application executing within development tool 118.

As will be developed in more detail, below, CDI 114 receives debug frames from an external application, and interlaces debug frames within the normal communication flow between computer system 102 and smart card 104. According to an exemplary embodiment, the debug frames are generated by a debug environment within application development tool 118 and sent to CDI 114. CDI 114 includes a debug filter to identify debug frames. The debug frames are generated within a unique identifiable attribute such as, for example, embedding an invalid source and/or destination address (e.g., FF hex) within the debug frame.

1 In addition, CDI 114 receives and identifies debug frames, i.e., response
2 debug frames, sent from smart card 104. The debug filter of CDI 114 identifies
3 the debug frames (e.g., by the invalid source/destination address) and promotes the
4 response to a debug environment, while normal application frames are promoted
5 to the application executing within application development tool 118. But for the
6 client development interface 114, computer system 102, applications 116 and
7 operating system 120 are each intended to represent any of a number of commonly
8 known computer systems, applications and operating systems, respectively, known
9 in the art.

10 According to one innovative aspect of the present invention, development
11 application 118 is intended to be any of a number of known software development
12 applications (also referred to as software development "tools"). Examples of such
13 software development tools include Visual Basic or Visual C/C++ from Microsoft
14 Corporation of Redmond, WA. Thus, a smart card application is developed using
15 computer 102 and a typical software development tool 118, utilizing the interlaced
16 debug protocol supported by CDI 114 and SCDI 110 to invoke and interrogate
17 smart card resources to verify the integrity of the developed code. By using
18 common software development tools such as those described above, the smart card
19 application development system 100 does not require the chip-specific, often
20 proprietary software development application and associated compilers, linkers
21 and debuggers that are typical of the cumbersome prior art development systems.

22 As alluded to above, the inclusion of the innovative CDI 114 and SCDI 110
23 within development system 100 support an interlaced debug protocol that
24 interlaces debug frames with standard application frames comprising a normal
25 communication flow between computer system 102 and smart card 104.

According to one embodiment, the debug frames are generated in response to user interaction with a debug environment of application development tool 118 executing a smart card application. The debug frames are sent to CDI 114, which identifies the debug frames and interlaces such frames with the normal application frames (generated by the application executing within the application development tool) and sent to smart card 104 via card reader 106 and communication medium 108. SCDI 110 receives the communication from computer system 102, identifies and routes the debug frames to a debug monitor, while application frames are promoted to an appropriate application/resource of the smart card (i.e., as identified by a source/destination address). The received debug frames include debug instructions which selectively invoke smart card resources (e.g., API's, device drivers, applications, etc.), providing a user with a heretofore unavailable view of system state information while an application is executing on the smart card. As described above, this state information is priceless during application development.

It should be appreciated that the interlaced debug protocol supported by the innovative CDI 114 and SCDI 110 of the present invention enables a user to employ an otherwise ordinary development tool 118 on an otherwise ordinary computer system to directly utilize the resources of smart card 104 to code and debug smart card applications. In addition to cost and ease of use advantages over the prior art, development system 100 represents a significant improvement over prior art development systems in that the applications developed using the present invention are easily ported from one smart card to another (i.e., the application is not chip specific, as is the case of systems developed using an ICE system). Moreover, insofar as the actual smart card resources are utilized during the

1 development, there is less of a chance for hidden bugs or other undetected
2 compatibility problems as measured against prior art development systems. In this
3 regard, application development system 100 represents a significant improvement
4 over the prior art in terms of cost, ease of use and quality of the end product.

5 Although the exemplary embodiment above discusses application
6 development and debugging using application development tool 118 and an
7 integrated debug environment, this is for ease of explanation only. An alternate
8 implementation may well provide an independent debug monitor executing on
9 computer 102, in communication with CDI 114 to interlace debug frames within
10 the normal communication flow generated by a host application, independently
11 executing on computer 102. Similarly, a debug application may well be embedded
12 within SCDI 110, or executing on smart card 104 as an independent entity. Thus,
13 the description above and below is to be regarded as merely illustrative, and not
14 limiting, of the spirit and scope of the present invention.

15 16 Example Computer System

17 In the discussion herein, the invention is described in the general context of
18 computer-executable instructions, such as program modules, being executed by
19 one or more conventional computers. Generally, program modules include
20 routines, programs, objects, components, data structures, etc. that perform
21 particular tasks or implement particular abstract data types. Moreover, those
22 skilled in the art will appreciate that the invention may be practiced with other
23 computer system configurations, including hand-held devices, personal digital
24 assistants, multiprocessor systems, microprocessor-based or programmable
25 consumer electronics, network PCs, minicomputers, mainframe computers, and

1 the like. In a distributed computer environment, program modules may be located
2 in both local and remote memory storage devices.

3 Fig. 2 shows a general example of a computer system 102 incorporating the
4 teachings of one aspect of the present invention, and suitable for use within the
5 smart card application development system 100. It will be evident, from the
6 discussion to follow, that computer 102 is intended to represent any of a class of
7 general or special purpose computing platforms which, when endowed with the
8 innovative client development interface 114, is suitable for use in smart card
9 application development system 100. In this regard, the following description of
10 computer system 102 is intended to be merely illustrative, as computer systems of
11 greater or lesser capability may well be substituted without deviating from the
12 spirit and scope of the present invention.

13 As shown, computer 102 includes one or more processors or processing
14 units 132, a system memory 134, and a bus 136 that couples various system
15 components including the system memory 134 to processors 132.

16 The bus 136 represents one or more of any of several types of bus
17 structures, including a memory bus or memory controller, a peripheral bus, an
18 accelerated graphics port, and a processor or local bus using any of a variety of
19 bus architectures. The system memory includes read only memory (ROM) 138
20 and random access memory (RAM) 140. A basic input/output system (BIOS) 142,
21 containing the basic routines that help to transfer information between elements
22 within computer 102, such as during start-up, is stored in ROM 138. Computer
23 102 further includes a hard disk drive 144 for reading from and writing to a hard
24 disk, not shown, a magnetic disk drive 146 for reading from and writing to a
25 removable magnetic disk 148, and an optical disk drive 150 for reading from or

writing to a removable optical disk 152 such as a CD ROM, DVD ROM or other such optical media. The hard disk drive 144, magnetic disk drive 146, and optical disk drive 150 are connected to the bus 136 by a SCSI interface 154 or some other suitable bus interface. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for computer 102. Although the exemplary environment described herein employs a hard disk 144, a removable magnetic disk 148 and a removable optical disk 152, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, random access memories (RAMs) read only memories (ROM), and the like, may also be used in the exemplary operating environment.

A number of program modules may be stored on the hard disk 144, magnetic disk 148, optical disk 152, ROM 138, or RAM 140, including an operating system 158, one or more application programs 160 including, for example, the innovative client development interface 114 or application development tool 118, other program modules 162, and program data 164. A user may enter commands and information into computer 102 through input devices such as keyboard 166 and pointing device 168. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are connected to the processing unit 132 through an interface 170 that is coupled to bus 136. A monitor 172 or other type of display device is also connected to the bus 136 via an interface, such as a video adapter 174. In addition to the monitor 172, personal computers often include other peripheral output devices (not shown) such as speakers and printers.

As shown, computer 102 operates in a networked environment using logical connections to one or more remote computers, such as a remote computer 176. The remote computer 176 may be another personal computer, a personal digital assistant, a server, a router or other network device, a network "thin-client" PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to computer 102, although only a memory storage device 178 has been illustrated in Fig. 2.

As shown, the logical connections depicted in Fig. 2 include a local area network (LAN) 180 and a wide area network (WAN) 182. Such networking environments are commonplace in offices, enterprise-wide computer networks, Intranets, and the Internet. In one embodiment, remote computer 176 executes an Internet Web browser program such as the "Internet Explorer" Web browser manufactured and distributed by Microsoft Corporation of Redmond, Washington to access and utilize online services.

When used in a LAN networking environment, computer 102 is connected to the local network 180 through a network interface or adapter 184. When used in a WAN networking environment, computer 102 typically includes a modem 186 or other means for establishing communications over the wide area network 182, such as the Internet. The modem 186, which may be internal or external, is connected to the bus 136 via a serial port interface 156. In a networked environment, program modules depicted relative to the personal computer 102, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

1 Generally, the data processors of computer 102 are programmed by means
2 of instructions stored at different times in the various computer-readable storage
3 media of the computer. Programs and operating systems are typically distributed,
4 for example, on floppy disks or CD-ROMs. From there, they are installed or
5 loaded into the secondary memory of a computer. At execution, they are loaded at
6 least partially into the computer's primary electronic memory. The invention
7 described herein includes these and other various types of computer-readable
8 storage media when such media contain instructions or programs for implementing
9 the innovative steps described below in conjunction with a microprocessor or
10 other data processor. The invention also includes the computer itself when
11 programmed according to the methods and techniques described below.
12 Furthermore, certain sub-components of the computer may be programmed to
13 perform the functions and steps described below. The invention includes such
14 sub-components when they are programmed as described. In addition, the
15 invention described herein includes data structures, described below, as embodied
16 on various types of memory media.

17 For purposes of illustration, programs and other executable program
18 components such as the operating system are illustrated herein as discrete blocks,
19 although it is recognized that such programs and components reside at various
20 times in different storage components of the computer, and are executed by the
21 data processor(s) of the computer.

22 Fig. 3 illustrates a block diagram of an example client development
23 interface (CDI) 114, suitable for use in computer system 102. As shown, CDI 114
24 comprises control logic 302, a debug filter 304 and memory 306, each coupled as
25 depicted. CDI 114 may well be characterized as an abstraction layer, enabling

1 higher level applications and resources of computer 102 to access and utilize
2 resources of smart card 104 endowed with SCDI 110 (which may also be
3 characterized as an abstraction layer). With respect to the present invention, the
4 features of CDI 114 are invoked upon detecting a debug frame by debug filter 304.
5 Debug filter 304 detects debug frames by analyzing each of the frames received by
6 the smart card for characteristics denoting a debug frame. According to one
7 implementation, debug filter 304 detects an invalid source and/or destination
8 address in a received frame marking the frame as a debug frame. It is to be
9 appreciated, however, that debug filter 304 can be configured to detect alternative
10 characteristics within a communication frame marking the frame as a debug frame.

11 Control logic 302 is intended to represent any of a broad range of logic
12 known in the art. In one implementation, control logic 302 is a processor, while in
13 alternate embodiments control logic is a microcontroller, a programmable logic
14 array, or a series of executable instructions which perform logic functions.
15 Control logic 302 communicates with smart card 104 in any of a plurality of
16 standard smart card communication protocols. In alternate embodiments, non-
17 standard protocols may well be used to communicate between controller 302 and
18 smart card 104 such as, for example, a unique development communication
19 protocol. Although not specifically denoted, it is to be appreciated that controller
20 302 communicates with smart card 104, and any other peripheral for that matter,
21 via the communication resources of operating system 120. Insofar as such
22 resources are well known in the art, they need not be described further here.

23 Upon detecting the arrival of a debug frame by debug filter 304, control
24 logic 302 marshals the debug frame parameters and interlaces the completed
25 debug frame with application frames comprising the normal communication flow

1 between computer 102 and smart card 104. According to one implementation,
2 control logic 302 analyzes the task to be performed by the received debug frame,
3 and places the debug frame at an appropriate point in the normal communication
4 flow for transmission to smart card 104 via card reader 106 and communication
5 medium 108.

6 Memory 306 includes one or more buffers wherein control logic 302
7 communication frames for transmission to smart card 104. In this regard, memory
8 306 is intended to represent any of a number of alternate memory devices
9 commonly known to those in the art.

10 Although depicted as a separate functional element, those skilled in the art
11 will appreciate that CDI 114 may well be integrated within and utilize the control
12 features associated with the application development tool 118. In one
13 implementation, for example, control logic 302 and memory 306 may well be
14 supplied by a debug environment within the application development tool 118,
15 wherein CDI 114 is comprised solely of debug filter 304. Accordingly, the
16 teachings of the present invention may well be practiced with variation from the
17 exemplary embodiment without deviating from the spirit and scope of the present
18 invention.

19 20 Example Smart Card

21 Fig. 4 illustrates a block diagram of an example IC card 104 suitable for
22 use within the application development system 100 of Fig. 1. In addition to the
23 innovative smart card development interface (SCDI) 110 and an operating system
24 with associated smart card resources 112, IC card 104 is shown comprising an
25 input/output interface 402, memory 406 having stored therein a plurality of

1 executable applications and/or applets 404, and control logic 408. As discussed
2 above, except for the inclusion of innovative smart card development interface
3 (SCDI) 110, to be described more fully below, smart card 104 is intended to
4 represent any of a broad category of integrated circuit (IC) cards commonly
5 known in the art. Thus, but for SCDI 110, each of I/O 402, applets 404, memory
6 406 and control logic 408 are likewise commonly known within the art and,
7 consequently, will not be further described here.

8 Fig. 5 illustrates a block diagram of an example SCDI 110, suitable for use
9 within any IC card such as, e.g., smart card 104 of application development
10 system 100 in Fig. 1. In accordance with the example implementation of Fig. 5,
11 SCDI 110 is shown comprising control logic 502, debug filter 504, debug monitor
12 506 and memory 508, coupled as depicted. According to one implementation,
13 SCDI 110 is invoked by CDI 114 upon receipt of a debug frame interlaced within
14 the normal communication flow between computer 102 and smart card 104.

15 According to this exemplary implementation, control logic 502 may be any
16 of a plurality of discrete logic and/or coded logic functions commonly known in
17 the art such as, for example, a processor, a controller, or a plurality of executable
18 instructions which implement such functionality. Similarly, memory 508 is
19 intended to represent any of a number of memory devices known in the art.

20 As above, debug filter 504 identifies debug frames within the received
21 communication flow by analyzing each of the frames received by the smart card
22 for characteristics denoting a debug frame. According to one implementation,
23 debug filter 504 detects an invalid source and/or destination address in a received
24 frame marking the frame as a debug frame. It is to be appreciated, however, that
25

1 debug filter 504 can be configured to detect alternative characteristics within a
2 communication frame marking the frame as a debug frame.

3 Debug monitor 506 selectively invokes one or more debugging features in
4 response to receipt of a debugging frame, as identified by debug filter 504.
5 According to one implementation, control logic 502 receives a debug frame, as
6 detected by debug filter 504, and promotes the debug frame to debug monitor 506.
7 Debug monitor controls and/or interrogates smart card resources (e.g., API's,
8 device drivers, applications, etc.) in accordance with the debug instructions
9 contained within the debug frame. As will be described more fully below, the
10 debug monitor 506 can read/write smart card memory 406, get/set breakpoints in a
11 smart card application 404, sequentially step a smart card application 404, run a
12 smart card application 404, release a smart card application 404, and obtain the
13 context of control logic 408. Although depicted as an element of SCDI 110,
14 debug monitor 506 may well be implemented as an independent debugging
15 application resident on smart card 104, without deviating from the spirit and scope
16 of the present invention.

17 18 Example Data Structures

19 Fig. 6 is a graphical illustration of a communication flow between a host
20 computer 102 and smart card 104 including one or more interlaced debug frames,
21 according to one aspect of the present invention. The innovative communication
22 flow 600 is shown comprising a plurality of application protocol data units
23 (APDU's) 602 and 606, selectively interlaced with one or more debug protocol
24 data unit (DPDU) 604 which enables an independent debug application to share
25 communication resources with a host application to control and interrogate the

005497-0400

otherwise closed architectural resources of a smart card. It is to be appreciated that, although depicted at the APDU level this is for ease of explanation only. That is, alternate embodiments are contemplated wherein interlacing of debug frames is performed at a block-level, i.e., at a lower level of communication. In this regard, it is to be appreciated that the DPDU is a construct of convenience to illustrate the features of the present invention. Alternate implementations at using lower-level or higher-level communication protocols are anticipated within the scope and spirit of the invention.

According to one implementation, the communication flow is a command/response communication protocol, initiated by the host system (e.g., computer 102). As alluded to above, the APDU's 602, 606 adhere to any of a number of accepted application protocols employed to communicate information between host applications and smart card applications. Accordingly, they need not be further described.

The DPDU 704 adheres to any of a number of accepted transport protocols used to communicate information between a smart card and a host system (e.g., a computer system). Fig. 6 breaks out DPDU 704 to denote a number of constituent fields including, but not limited to, a node address field 608, a protocol control block 610, a length field 612, a data field 614 and an error detection field 616. It is to be appreciated that the illustrated size of the fields do not necessarily correspond to relative block sizes, and the order of the fields may well be changed without deviating from the spirit and scope of the present invention.

The node address field 608 contains source and/or destination address information. More specifically, the node address field 608 includes information regarding the virtual address of the application issuing the frame (source address)

1 and the address of the application to which the frame is being sent (destination
2 address). According to one aspect of the invention, a debug frame is denoted as
3 such by embedding invalid source and/or destination addresses in the node address
4 field 608. Upon receiving a communication flow, debug filter (304, 504)
5 identifies the invalid source and/or destination address and routes the debug frame
6 to a debug application or monitor.

7 The protocol control block 610 denotes whether the frame is a command
8 frame or a response frame, and whether the received frame is the last frame in a
9 communication, or whether more frames follow to deliver the data required to
10 complete the communication instance.

11 As their names imply, the length field 612 provides an indication as to the
12 length of the frame, while the data field 614 carries the instructions/data
13 communicated between the host and the smart card. According to one
14 implementation, the data field 614 includes debug instructions that direct a debug
15 monitor 506 of a SCDI 110 to perform some task. Examples of such debug
16 instructions (in pseudo code) and their effect include:

17 Debug (run app_x, Data) - execute an application using Data

18 Debug (get context) - obtain the context of smart card control logic

19 Debug (read memory) - read smart card memory

20 Debug (step) - step an application executing on the smart card.

21 Debug (set breakpoint) - set a breakpoint within an application
22 executing on the smart card.

23 Debug (run) - execute application on smart card until event
24 (e.g., breakpoint)
25

1 Additional debug instructions and their function can be found within the Appendix
2 attached hereto.

3 The error checking field 616 of DPDU 604 includes information utilized by
4 a development interface (i.e., CDI 114 or SCDI 110) to verify the integrity of the
5 received frame. Any of a number of suitable error checking schemes may well be
6 used such as, for example, inclusion of a checksum.

7 8 9 Example Operation

10 Fig. 7 is a flow chart of an example method for debugging an IC card
11 application using an interlaced debug protocol, according to one aspect of the
12 present invention. For ease of explanation, and not limitation, the method of Fig.
13 7 will be developed with continued reference to Fig.'s 1-6.

14 Turning to Fig. 7, the method begins with step 702 wherein the debug
15 environment of a host computer 102 is invoked. In one implementation, the debug
16 environment resides within application development tool 118, while in alternate
17 embodiments, the debug environment is a stand-alone application 116.

18 In step 704, a debug command frame is generated within the debug
19 environment. More specifically, a user instructs the debug environment to invoke
20 a debug feature of coupled smart card 104. As described above, the debug
21 environment marks the debug frame as such utilizing an invalid address in node
22 address field 608 of the generated debug frame (e.g., DPDU 604). Once
23 generated, the debug frame is sent to the CDI 114.

24 The CDI 114 receives the debug frame and interlaces the received debug
25 frame with other application frames comprising the normal communication flow,

1 application 404 and/or to interrogate smart card resources. In response to
2 execution of received communication frames (i.e., application and debug) within
3 their respective applications (i.e., smart card application and debug monitor),
4 smart card application 404 may generate response application frames. Similarly,
5 debug monitor 506 may generate response debug frames depending, of course, on
6 the point at which execution of the applications is suspended, step 716. In this
7 manner, debug monitor 506 of SCDI 110, in response to the innovative interlaced
8 debug protocol, manage execution of smart card applications, and disclosure of
9 smart card state information to facilitate application development.

10 In step 718, SCDI 110 receives the response application frames generated
11 by smart card application 404 and interlaces received response debug frames
12 generated by debug monitor 506, if any, to generate a response communication
13 flow. The communication flow is transmitted to the computer system 102 via
14 communication medium 108 and card reader 106, step 720.

15 In step 722, CDI 114 receives the response communication flow from the
16 smart card 104. More specifically, debug filter 304 receives and analyzes the
17 response frames to detect debug frames. In step 724, controller 302 of CDI 114
18 promotes application frames to the host application 116, while identified debug
19 frames are promoted to the debug environment.

20 Turning briefly to Fig. 8, a signaling diagram for an example
21 communication session between a host system and a smart card utilizing the
22 interlaced debug protocol of the present invention is presented. As shown, the
23 communication is broken down according to the functional elements discussed
24 above, namely, between application development tool 118, a debug environment
25 802, CDI 114, SCDI 110, debug monitor 506 and a smart card application 404.

